Performance Characterization of the Dynamic Programming Obstacle Detection Algorithm

Tarak Gandhi, *Member, IEEE*, Mau-Tsuen Yang, Rangachar Kasturi, *Fellow, IEEE*, Octavia I. Camps, *Member, IEEE*, Lee D. Coraor, *Member, IEEE*, and Jeffrey McCandless

***Abstract*—A computer vision-based system using images from an airborne aircraft can increase flight safety by aiding the pilot to detect obstacles in the flight path so as to avoid mid-air collisions. Such a system fits naturally with the development of an external vision system proposed by NASA for use in high-speed civil transport aircraft with limited cockpit visibility. The detection techniques should provide high detection probability for obstacles that can vary from subpixels to a few pixels in size, while maintaining a low false alarm probability in the presence of noise and severe background clutter. Furthermore, the detection algorithms must be able to report such obstacles in a timely fashion, imposing severe constraints on their execution time. For this purpose, we have implemented a number of algorithms to detect airborne obstacles using image sequences obtained from a camera mounted on an aircraft. This paper describes the methodology used for characterizing the performance of the dynamic programming obstacle detection algorithm and its special cases. The experimental results were obtained using several types of image sequences, with simulated and real backgrounds. The approximate performance of the algorithm is also theoretically derived using principles of statistical analysis in terms of the signal-to-noise ration (SNR) required for the probabilities of false alarms and misdetections to be lower than prespecified values. The theoretical and experimental performance are compared in terms of the required SNR.**

***Index Terms*—Autonomous navigation, dynamic programming, obstacle detection, performance characterization, target detection, .**

I. Introduction

CONTINUED advances in the fields of image processing and computer vision have raised interest in their suitability to aid pilots to detect possible obstacles in their flight paths. For the last few years, NASA has been exploring the use of image sequences for detecting obstacles in the flight path of an aircraft. In the design of a high-speed civil transport (HSCT) aircraft with a limited cockpit visibility, NASA has proposed an external visibility system (XVS) in which high-resolution video images would be obtained using cameras mounted on the aircraft. These images can be used to detect obstacles in the flight path to warn the pilots and avoid collisions.

Algorithms for the detection of airborne objects from images are abundant in the published literature. Nishiguchi *et al.* [10] proposed the use of a recursive algorithm to integrate multiple frames while accounting for small object motion. A dynamic programming approach was used by Barniv [2] and Arnold *et al.* [1] to detect moving objects of small size.

Although the algorithms claim good performance, most of them have been tested only under very restrictive image input sets. A common problem that researchers in computer vision must face is that algorithms often fail to perform as required when they are presented with inputs different from the ones that had been previously tested with. Most researchers circumvent this problem by tuning up parameters in some *ad-hoc* manner or by designing yet another algorithm tailored for the particular application. Unfortunately, this approach requires a very expensive trial and error process that needs to be repeated whenever the problem specifications or the inputs change.

In this paper, we present the methodology we have used for systematic characterization of the performance and limitations of obstacle detection algorithms [4], [9]. In particular, we have characterized the performance of the dynamic programming target detection algorithm [1], [2] and compared its performance with temporal averaging and single frame thresholding, which can be expressed as special cases of the dynamic programming algorithm. The performance is characterized experimentally by modeling the image degradation and applying the algorithms to simulated and real image sequences. A methodology adapted from psychology literature is used as in [8] to condense the information obtained from the performance characterization curves. Using statistical methods as in [11], the theoretical performance of the detection algorithms is derived in terms of the signal-to-noise ration (SNR) required to limit the rate of false alarms and misdetections.

II. Performance Characterization Methodology

Consider a detection algorithm that must report whether a given image has a target or not. Typically, the algorithm would compute some measure of evidence of target presence and compare it to some given threshold value. Whenever the evidence measure is greater than the given threshold, a target would be

Manuscript received April 18, 2000; revised October 21, 2003. This work was supported in part by the NASA Ames Research Center under Grant NAG2-1152. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Thiow Keng Tan.

T. Gandhi is with the Computer Vision and Robotics Research Laboratory, University California at San Diego, La Jolla CA 92093-0434 USA (e-mail: tarakgandhi@hotmail.com).

M.-T. Yang is with the Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien, Taiwan, R.O.C. (e-mail: mtyang@mail.ndhu.edu.tw).

R. Kasturi is with the Department of Computer Science and Engineering, University of South Florida, Tampa, FL 33620 USA (e-mail: r1k@cse.usf.edu).

O. I. Camps is with the Department of Electrical Engineering and the Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA 16802 USA (e-mail: camps@psu.edu).

L. D. Coraor is with the Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA 16802 USA (e-mail: coraor@cse.psu.edu).

J. McCandless is with the NASA Ames Research Center, Moffett Field, CA 94035 USA (e-mail: jeffrey.mccandless@nasa.gov).

Digital Object Identifier 10.1109/TIP.2005.863973

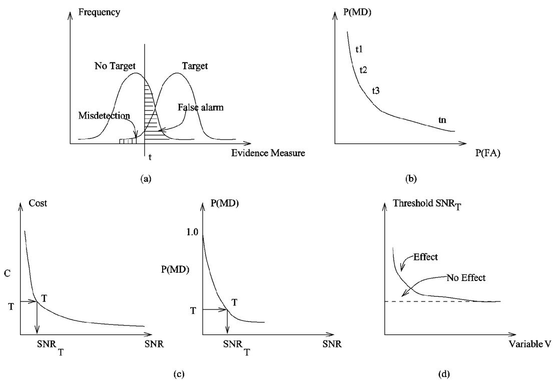


Fig. 1. Steps for performance characterization. (a) Step 1: Obtain the frequency distributions of the evidence measure for images with and without target. (b) Step 2: Obtain the ROC. (c) Step 3: Determine the optimal operating point using either the expected cost or the probability of detection given the probability of false alarm. (d) Step 4: Plot the threshold value corresponding to the optimal operating point versus a variable of interest.

reported. The performance of the algorithm is usually characterized by the "receiver operating curve" (ROC), where the probability of misdetection is plotted versus the probability of false alarm as some tuning parameter is changed. The performance of the algorithm is affected by several factors, such as image contrast, target size, complexity of the background, etc. The effect of variations of these variables on the overall performance can be measured through the use of equivalent effects of some critical signal variable by following the methodology proposed by Kanungo *et al.* [8] as summarized in the four steps described as follows.

1) *Obtain evidence distributions:* The first step consists on estimating distributions of evidence measures, one for images with target and another for images without target, as illustrated in Fig. 1(a). This estimation is done nonparametrically by randomly presenting the algorithm with images of both types and recording the frequency of the evidence measure values reported by the algorithm, using a histogram.
2) *Obtain ROCs:* The second step consists on constructing an ROC as the one shown in Fig. 1(b) by varying the threshold used by the algorithm to compare against the computed evidence measure. False alarms occur when a pixel in the given image does not contain a target, but the evidence measure is greater than the threshold being used. Misdetections occur when the given image contains a target, but the evidence measure is less than the threshold. The probabilities of false alarms and misdetections can be approximated by their frequency ratios

$$P(\text{FA}) = P(H_1|H_0) = \\ = \frac{\text{Number of false alarms}}{\text{Total number of pixels without target}}$$
$$P(\text{MD}) = P(H_0|H_1) = \\ = \frac{\text{Number of mis-detections}}{\text{Total number of targets in input images}}$$

where H_0 and H_1 denote the hypotheses corresponding to the absence and presence of a target, respectively.

3) *Determining the optimal operating point:* The optimal operating point (or its corresponding threshold value) can be specified in different ways, depending on how much prior knowledge is available. If the prior probabilities and costs are known, the optimal operating point can be defined as the one minimizing the expected cost. Let C_{10}, C_{01}, C_{11}, and C_{00} be the costs of a false alarm, a misdetection, a correct detection, and a correct rejection, respectively. The expected cost is then given by

$$E[C] = [P(H_0|H_0)C_{00} + P(H_1|H_0)C_{10}]\,P(H_0) \\ + [P(H_0|H_1)C_{01} + P(H_0|H_1)C_{11}]\,P(H_1). \quad (1)$$

The optimal operating point is found by minimizing $E[C]$ with respect to the threshold to be used by the algorithm. In the most likely case when the costs are difficult to set, an alternative way to define the required operating point is to use the Neyman–Pearson criterion—i.e., to maximize the probability of detection for a *given* probability of false alarm.

Independently of which definition is used, the optimal operating point depends on the SNR in the input image. For example, increasing the target contrast results in an increase of the SNR and, hopefully, in an improvement of the algorithm performance for a given threshold value. The optimal operating points for different SNRs can be found by repeating steps 1 and 2 for the corresponding SNR values and determining the optimal point for each of the resulting ROCs. Once this is done, a graph of the expected cost or the probability of detection versus SNR can be plotted, depending on which definition of operating point is being used. This is illustrated in Fig. 1(c). Finally, let SNR_T and T be the SNR and the associated threshold values for the optimal operating point for a given level of performance, as shown in the figure. The level of performance is specified by either a desired expected cost of classification or a desired probability of misdetection, again, depending on which optimal criterion is used.
4) *Analysis with respect to variables of interest:* Besides SNR, other factors affect the algorithm performance and merit study. Examples are the size of the target, the amount of target motion on the images, and the amount and nature of image clutter. In order to study these effects, steps 1)–3) are repeated for different values of variables representing these variations. These results are then summarized in a graph where the threshold T determined in step 3 is plotted against the value of the variable of interest, as shown in Fig. 1(d). A fairly flat plot indicates that the effect of the variable being considered on the optimal operating point of the algorithm is negligible. On the other hand, a steep plot indicates that the variable has a high impact on the performance.

It should be noted that a smaller SNR threshold T implies better performance, since weaker targets can be detected with the same given rates of false alarms and misdetections. Measuring the performance in terms of the SNR threshold makes it easier to measure and compare the performance of different algorithms, or the same algorithm with different parameters. This is because the variables, such as the false alarm and misdetection rates are eliminated from the curves, making place for other parameters.

III. Dynamic Programming Algorithm

In the following sections, the performance of the dynamic programming algorithm will be characterized using the above methodology. This algorithm performs temporal integration of the signal strength over a number of image frames, thus increasing the effective SNR.

Before the dynamic programming algorithm can be applied, preprocessing should be performed on the input images to suppress the background. In the case of an image with little or no clutter, a low-stop filter subtracting from every pixel, the local average of the neighborhood of that pixel, can be used to this effect. However, if the background has significant clutter, the low-stop filter is not as effective removing it. A morphological filter implemented by subtracting the morphological opening from the morphological closing of the image is more effective in removing clutter [3]. A simple measure of clutter that could be used to decide which filter to apply is the pixel variance in the neighborhood of each pixel.

To account for the motion of the target, the algorithm discretizes the velocity space into cells. Temporal integration is performed separately for each cell, assuming that the target is moving in that particular direction. Since the velocity of the target could be arbitrary, the velocity space (u, v) is discretized within the range of possible target velocities. A set of intermediate images F, each corresponding to a particular velocity (u, v), are created recursively using the following steps.

1) *Initialization:* For all pixels (x, y) and all velocities (u, v), set

$$F(x, y; u, v; 0) = 0. \tag{2}$$

2) *Recursion:* At time k, set

$$\begin{aligned} F(x, y; u, v; k) = {} & f(x, y; k) \\ & + \alpha \max_{(x', y') \in Q} F(x - u - x', y - v - y'; u, v; k - 1) \end{aligned} \tag{3}$$

where $Q = \{(x', y') | x'_{\min} \leq x' \leq x'_{\max}, y'_{\min} \leq y' \leq y'_{\max}\}$.

3) *Termination:* At time K, take

$$F_{\max}(x, y; K) = \max_{(u,v) \in P} F(x, y; u, v; K) \tag{4}$$

where $P = \{(u, v) | u_{\min} \leq u \leq u_{\max}, v_{\min} \leq v \leq v_{\max}\}$.

The maximum operation in the recursion step is performed using the set Q, which ensures that the targets with velocities which do not fall on the grid are not missed. The set of discretized velocities denoted by P determines the range of target velocities that can be detected by the algorithm. The final maximum in the termination step combines the targets corresponding to all the velocities. The number of elements in P and Q are denoted by p and q, respectively.

In the recursion step, a maximum is taken over q pixels. If these pixels are all noise pixels, they are more likely to give a false alarm if q is large. Thus, the rate of false alarms increases with q. To get better performance, a smallest possible q should be used. The value of $q = 4$ has been used in our experiments corresponding to a 2×2 neighborhood, given by

$$Q = \{(0, 0), (-1, 0), (0, -1), (-1, -1)\}. \tag{5}$$

This ensures that the targets having fractional velocities are not missed. The asymmetry in this neighborhood is compensated by choosing $u_{\min} = u_{\max} - 1$ and $v_{\min} = v_{\max} - 1$. For the case of $u_{\max} = v_{\max} = 1$, $p = 4$ and P is given by

$$P = \{(0, 0), (1, 0), (0, 1), (1, 1)\}. \tag{6}$$

The algorithm then detects targets with a maximum velocity of 1 pixel per frame. However, when spatial integration is performed prior to dynamic programming using image pyramid approach, targets with larger sizes and velocities can also be detected.

On the other hand, if $P = Q = \{(0, 0)\}$ so that $p = q = 1$, the algorithm reduces to recursive temporal averaging, which gives the best performance for stationary targets. However, the performance of temporal averaging sharply degrades if the target is moving, whereas that of dynamic programming algorithm does not.

The output of the dynamic programming algorithm is an image, with large values at positions where the target strength is high. However, the pixels in the neighborhood of the target will also have a significantly large value. This can be resolved by using nonmaximal suppression, where the output is smoothed using a Gaussian filter of standard deviation 1, and each pixel which is not a local maximum in its 3×3 region is set to zero. After this, only the pixels which are local maxima remain.

Fig. 2(a) shows the last image from a sequence of synthetic images containing a single moving target, immersed in Gaussian noise. Fig. 2(b) shows the target trajectory, with the blob marking the end. The output of the dynamic programming algorithm before nonmaximal suppression is shown in Fig. 2(c). It is seen that the target is dilated, i.e., there is a significant output not only at the target position, but also around it. Non-maximal suppression removes this dilation, and the final output is shown in Fig. 2(d). Notice that there are a number of potential (false) targets in addition to the actual target, that might be eliminated by thresholding this image.

IV. Experimental Protocol

In this section, the experimental protocol used to characterize the performance of the target detection algorithms, is described in detail. The protocol consists of the following components, specifying how to

1) generate images of simulated targets;
2) apply the detection algorithm;
3) estimate the rates of false alarms and misdetections (ROCs) for different sets of parameters;
4) characterize the algorithm performance by condensing the ROCs into a performance curve.

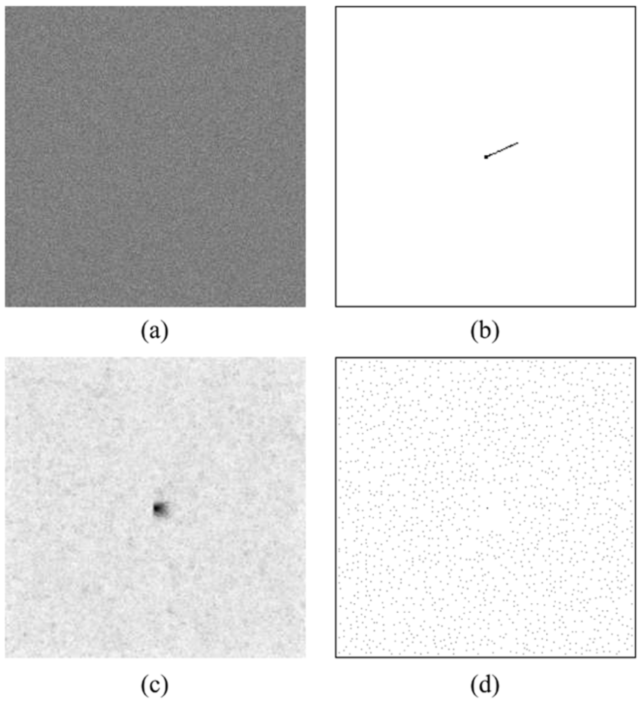


Fig. 2. (a) Last image from a sequence of images containing a single moving target. (b) Trajectory of the target, with the blob marking the end. (c) Output of the dynamic programming algorithm before nonmaximal suppression. The target gets dilated. (d) Output of nonmaximal suppression which removes this dilation. Notice that there are a number of potential (false) targets in addition to the actual target, that might be eliminated by thresholding this image.

A. Image Generation

In order to characterize the performance of the detection algorithm, it is applied to sequences of synthetic images with and without targets. While the images with targets are used to estimate the misdetection rate, the images without targets are used to estimate the false alarm rate. The images can have the following different types of backgrounds.

1) *Synthetic noise from camera model:* The background is assumed to have a constant value A_{bg}. The noise is artificially simulated, using the camera noise model.
2) *Real noise from a digital camera:* The background images are taken from a sequence of images obtained from a digital camera looking at a scene with constant intensity such as clear sky.
3) *Real background from an analog camera:* The background images are obtained using a sequence of images with significant clutter. The sequence[1], which was provided by NASA, was captured using an analog camera mounted on a flying aircraft. Fig. 3 shows a typical frame of this sequence.

1) Generation of Image Sequences: To estimate the number of false alarms, the background images themselves, without any addition of targets are used directly. The size of these images is $N_x \times N_y$. For estimation of the rate of misdetections, simulated targets are inserted in the background images generated as described below. For each simulation, a target file is created having information on the position, velocity, size, amplitude, and each target to be placed in an image. The image size is taken as $N_x \times N_y$. The number of targets to be inserted in every image is N_{targ}. The target trajectories are generated in such a way that the detection of one target does not interfere with the detection of another. This is accomplished by drawing a window around each target trajectory. The next generated trajectory is valid only if the window around it does not overlap with the windows around the previously generated targets. Otherwise, the procedure is repeated by generating another trajectory, until the total number of valid trajectories is N_{targ}.

[1]Unfortunately, due to the risks involved while flying near other aircrafts, only limited data of this type is available.

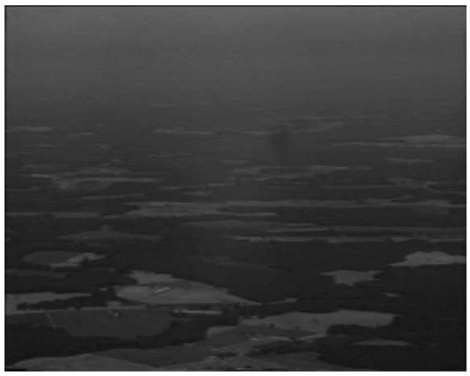


Fig. 3. Sample image from the real background sequence provided by NASA. The image sequence was taken from an analog camera mounted on an aircraft.

The velocity (V_x, V_y) of the targets is uniformly distributed so that $-u_{\max} \le V_x \le u_{\max}$ and $-v_{\max} \le V_y \le v_{\max}$. The position of the targets is specified for the *last* frame—i.e., when the detection is completed. The position of the target in other frames is given by $(x - V_x\Delta t, y - V_y\Delta t)$, where Δt is the time-interval between the given frame and the last frame.

A target can be a point target, or have a specified height and width. The size of the target is given by $s_x \times s_y$. The target amplitude is given by A. For point targets, the amplitude corresponds to the contrast of the pixel it occupies, with respect to the background. However, for an extended target, the contrasts of all the occupied pixels are given by the product of the target amplitude and the fraction of the area in the respective pixel that is covered by the target.

Fig. 4(a) shows the trajectories of simulated targets to be added to an image, and Fig. 4(b) shows a zoomed part on a portion of the image. The end of the trajectories are marked by blobs. The black box around the target denotes the region where another target cannot be present, to reduce the interference between the targets.

Once the file describing the targets is created, an image sequence of N_{frame} frames is generated. For each frame, the position of the targets are calculated, and the targets are inserted accordingly. For point targets, the amplitude is added to the background image in the target position pixel. For extended targets occupying a number of pixels (fully or partially), the product of the amplitude and the fractional occupancy is added to the background image at that pixel.

2) Addition of Noise: Two types of camera noise [7], [9], the fixed-pattern noise (FPN) and the temporal noise, are added

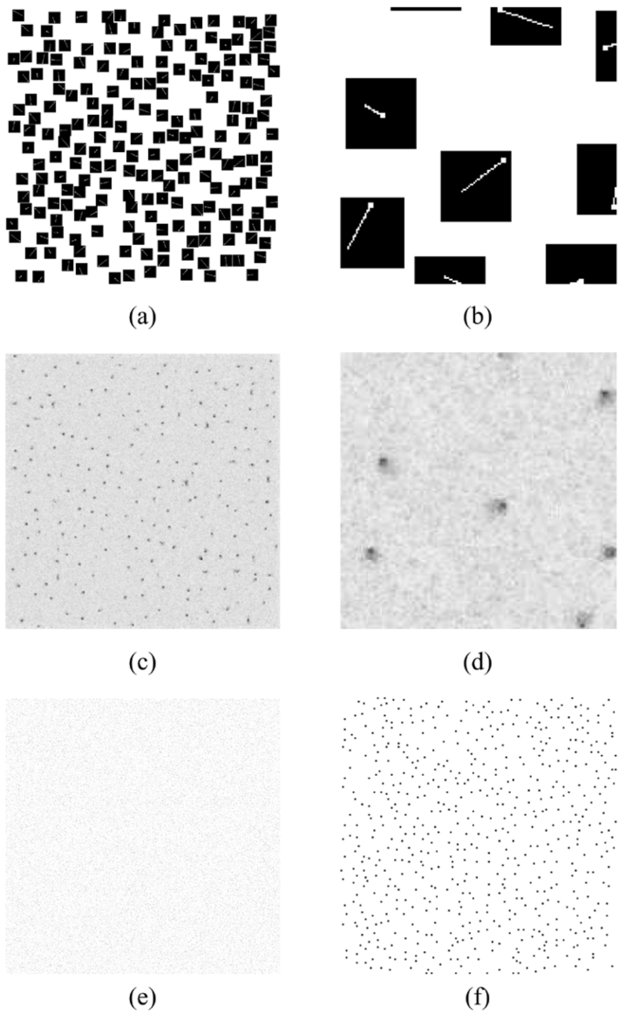


Fig. 4. Detection using dynamic programming. (a) Simulated targets trajectories. There are 200 targets, and the image size is 960 × 960. The end of the trajectory is marked by a blob. The targets are separated so that there the interference between them is reduced. The black box around the target denotes the region where another target cannot be present. (b) Zoomed part of the target trajectory image. (c) Dynamic programming output of a typical experiment (before nonmaximal suppression). (d) Zoomed part of the output. (e) Dynamic programming output of the same experiment without adding targets—i.e., false alarms. (f) Zoomed part of the output.

to the sequences created using synthetic backgrounds. FPN has two components, additive, and multiplicative. The parameters of this noise change from pixel to pixel, but do not change with time. The parameter values for each pixel are determined a priori using the camera, and stored as images. On the other hand, the temporal noise is completely random, and is generated separately for each frame. The temporal noise approximately follows a Gaussian distribution with a variance

$$\sigma^2_{\text{temp}} = w_0 + w_1 I$$

where I is the expected gray value of the pixel, and w_0, w_1 are the parameters of the particular camera. However, since the background amplitude A_{bg} is constant for the experiments with simulated noise, and the target amplitude $A \ll A_{bg}$, we have $I = A + A_{bg} \simeq A_{bg}$ and σ^2_{temp} is approximately constant, given by $\sigma^2_{\text{temp}} \simeq w_0 + w_1 A_{bg}$. Using the approach similar to Healey and Kondepudy [6], the values of the parameters for the particular camera were estimated [9] as $w_0 = 0.171$ and $w_1 = 0.0056$. For background $A = 128$, this gives $\sigma^2_{\text{temp}} = 0.888$. The image is then quantized to give the output in byte format. This adds a quantization noise corresponding to $\sigma^2_{\text{quant}} = 1/12$. Neglecting the cross-correlation the total noise would have $\sigma^2_{\text{noise}} = 0.971$ or $\sigma_{\text{noise}} = 0.986$.

B. Algorithm Application

The target detection algorithm whose performance is to be characterized is applied to each simulated image sequence. In the cases of synthetic images, and digital camera sequences, FPN can be corrected in advance by using precomputed parameters of FPN for each pixel. These parameters are perturbed by a random amount corresponding to their estimated covariance, to model the error in estimating these values. Experiments are performed without and with correction of FPN, and the results are compared.

According to the type of background used, preprocessing in the form of a low-stop filter or a morphological filter are performed before applying dynamic programming. After dynamic programming is applied, nonmaximal suppression is performed to ensure correct counting of false alarms and misdetections. The outputs of a typical experiment with 200 targets is shown in Fig. 4(c)–(f).

C. Estimation of False Alarms and Misdetections

The algorithm to be characterized is applied on the image sequences with as well as without targets. The sequences without targets are used to estimate the false alarm rate, whereas the sequences with targets are used to estimate the misdetection rate.

For the false alarm rate, the histogram of the output image is obtained. Using this histogram, the false alarm rates for different thresholds can be obtained. For the misdetection rate, only the pixels in a specified window of 5×5 pixels around the specified target position are checked. For each such window corresponding to a single target, the maximum value of the algorithm output is taken. A histogram of these maximum values is formed, and processed to obtain the misdetection rates for different thresholds. The false alarm and misdetection rates are averaged over a number of simulations N_{FA} and N_{MD}, respectively.

The number of simulations to test can be specified so that the standard deviation in the estimate of the false alarm or misdetection rate is below a given value. This can be seen by observing that the occurrence of an event such as a false alarm or a misdetection can be modeled as a Poisson process, and, therefore, the variance of the total number of events is equal to the mean. Thus, if n events are observed, the standard deviation of the absolute error in the number of events is $\sqrt{n}$, and that of the relative error is $1/\sqrt{n}$. For example, for $n = 10$ events, the error standard deviation is 3.2, or 32% of the number of events. This error estimate can be confirmed by measuring the variance of these rates across the simulations.

D. Performance Characterization

Using the estimated false alarm and misdetection rates, the ROC can be plotted showing the rate of misdetection against

TABLE I
PARAMETERS USED FOR THE EXPERIMENTS WITH THE FOLLOWING IMAGE CATEGORIES: (1) SYNTHETIC NOISE FROM CAMERA MODEL, (2) REAL NOISE FROM A DIGITAL CAMERA, AND (3) REAL BACKGROUND FROM AN ANALOG CAMERA

Description	Parameter	Category		
		(1)	(2)	(3)
Img. x size	N_x	960	960	640
Img. y size	N_y	960	960	480
No. of targets	N_{targ}	200	200	50
Max. x velocity	u_{max}	1	1	1
Max. y velocity	v_{max}	1	1	1
x size	s_x	0.5 – 2	2	2
y size	s_y	0.5 – 2	2	2
Amplitude	A	1.0 – 15.0	1.0 – 6.0	10.0 – 70.0
No. of frames	N_{frame}	32	32	32
Background value	A_{bg}	128	$\simeq$ 200	–
Noise std. dev.	σ_{noise}	0.942	–	–
Forgetting factor	α	15/16	15/16	15/16
No. of FA sims.	N_{FA}	500	1	1
No. of MD sims.	N_{MD}	50	10	10
Thrs. FA rate	FA_T	0.02	10	10
Thrs. MD rate	MD_T	0.001	0.01	0.01

the rate of false alarms. The misdetection rate for a specified false alarm rate (FA_T) is noted from the curve. The simulations are repeated for a number of signal amplitudes A. The ratio of this amplitude to noise level σ_{noise} corresponds to the SNR. The value of the signal amplitude for a specified misdetection rate (MD_T), and the above false alarm rate is obtained. This is considered as the threshold signal value (A_T). The number of simulations used is at least $N_{\mathrm{FA}} = 10/\mathrm{FA}_T$ in the case of false alarms and $N_{\mathrm{MD}} = 10/\mathrm{MD}_T$ in the case of misdetections, so that for the rates FA_T and MD_T, an average of at least ten events would be observed, giving an error σ of at most 32%. Due to constraints on the execution time, larger number of experiments were not used, although they would be desirable for reducing this error.

Other parameters, such as the size of the target, can be varied one at a time, and the variation of A_T can be plotted against the respective parameter to determine the effect of the parameter on the algorithm performance.

V. EXPERIMENTAL RESULTS

The target detection algorithm was tested on three categories of images, as described in the protocol, using the parameter values listed in Table I. The results are shown and compared in the following section.

A. *Synthetic Noise From Camera Model*

In this case, the noise was synthetically generated using the noise model of the Kodak Megaplus ES 1.0 digital camera. Targets of varying size were added for misdetection analysis. Experiments without and with correction of FPN were performed.

Fig. 5(a) and (b) shows the plots of the false alarm and misdetection rates, respectively, against the threshold value, for experiments without FPN correction. The misdetection rates are shown for a number of signal amplitudes for 1×1 targets. The misdetection rate is measured as the ratio of the average number of misdetections, to the total number of targets in a simulation. However, the false alarm rate is measured as the average number of false alarms per simulation, instead of the ratio of the number

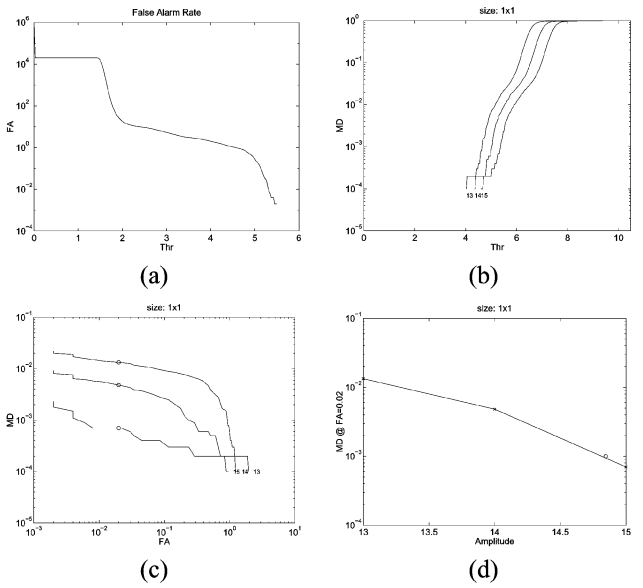


Fig. 5. Results for camera noise model without FPN correction. (a) Plot of FA rate (average number per simulation) against threshold. (b) Plot of MD rate against threshold, for a number of signal amplitudes (higher amplitudes toward right) for 1×1 targets. (c) Plot of MD rate against FA rate (for marked amplitude). The data points are marked as crosses. The MD rate when FA rate is $\mathrm{FA}_T = 0.02$ per simulation is interpolated, and plotted as circle. (d) Plot of MD rate against amplitude for FA rate of $\mathrm{FA}_T = 0.02$ per simulation. The value amplitude when MD rate is $\mathrm{MD}_T = 0.001$ per target is interpolated and marked as a circle.

of false alarms to the total number of pixels. This is done to give a better idea of the algorithm performance.

Fig. 5(c) shows the plot of misdetection rate against false alarm rate for different amplitude values for 1×1 targets. The point of threshold false alarm rate FA_T is set to 0.02 false alarms per simulation, which corresponds to a total of ten false alarms for $N_{\mathrm{FA}} = 500$ simulations. Fig. 5(d) shows the plot of misdetection rate against the amplitude values for the above rate of false alarms. The A_T for the threshold misdetection rate of MD_T is interpolated, and marked as a circle. The MD_T is set to a probability of 0.001 per target, which corresponds to an average of 0.2 misdetections per simulation for a simulation with 200 targets, or a total of ten misdetections for $N_{\mathrm{MD}} = 50$ simulations. The corresponding graphs for the case where fixed pattern noise compensation was applied are shown in Fig. 6.

The above experiments are repeated for other sizes of targets, and the A_T calculated from these is plotted against the size of the target. Resulting plots for the experiments without FPN correction are shown in Fig. 7(a) for square targets (size $x \times x$) and in Fig. 7(b) for rectangular targets (size $1 \times x$). The corresponding results for the experiments with FPN correction are shown in Fig. 7(c) and (d). The threshold amplitudes for various sizes are tabulated in Table II. It is seen that larger targets require smaller signal amplitudes for detection implying better performance. Similarly, the signal amplitudes required when FPN correction is applied are much smaller than those when the correction is not applied, implying better performance in the former case.

B. *Real Noise From a Digital Camera*

In this case, instead of synthetically generating the noise, background images captured using the Kodak Megaplus ES 1.0

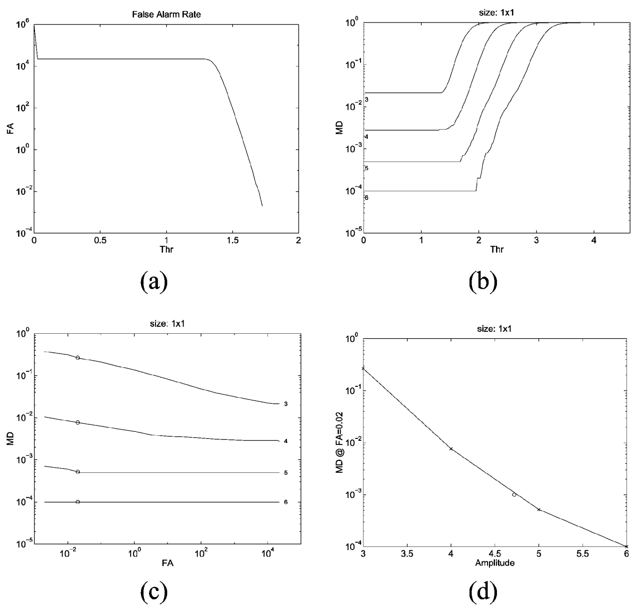


Fig. 6. Results for camera noise model with FPN correction. (a) Plot of FA rate (average number per simulation) against threshold. (b) Plot of MD rate against threshold, for a number of signal amplitudes (higher amplitudes toward right) for 1×1 targets. (c) Plot of MD rate against FA rate (for marked amplitude). The data points are marked as crosses. The MD rate when FA rate is $\mathrm{FA}_T = 0.02$ per simulation is interpolated, and plotted as circle. (d) Plot of MD rate against amplitude for FA rate of $\mathrm{FA}_T = 0.02$ per simulation. The value amplitude when MD rate is $\mathrm{MD}_T = 0.001$ per target is interpolated and marked as a circle.

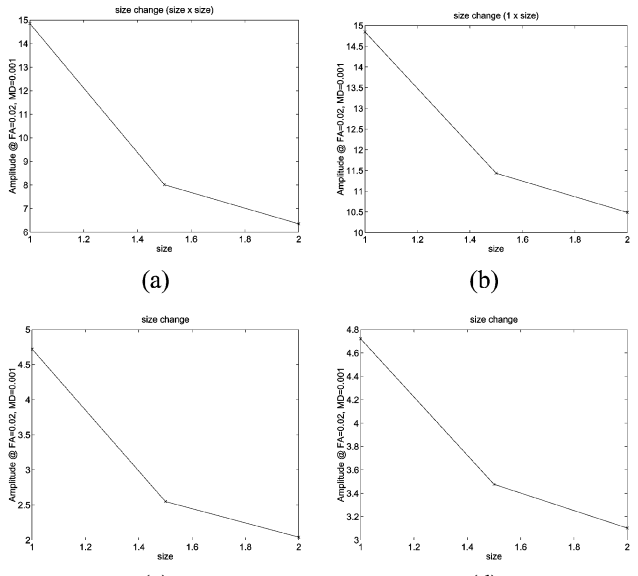


Fig. 7. Performance curves for simulated targets. (a) Plot of amplitude against the target size $(x \times x)$ for experiments without FPN correction. The data points are marked as crosses. (b) Plot of amplitude against the target size $(1 \times x)$. (c) and (d) Corresponding plots for experiments with FPN correction.

digital camera looking at the sky were used. Targets of size 2×2 pixels were synthetically added for the misdetection analysis. Experiments without and with correction of FPN were also performed. The false alarm threshold was set $\mathrm{FA}_T = 10$ per simulation, resulting in a total of ten false alarms for $N_{\mathrm{FA}} = 1$ simulation. The misdetection threshold was set to $\mathrm{MD}_T = 0.01$ per target, corresponding to 20 misdetections for $N_{\mathrm{MD}} = 10$ simulations with $N_{\mathrm{targ}} = 200$ targets. As shown in Section VI,

TABLE II
RESULTS OF DYNAMIC PROGRAMMING ALGORITHM ON SIMULATED IMAGE SEQUENCES WITHOUT AND WITH FPN CORRECTION. THRESHOLD AMPLITUDES ARE SHOWN FOR FALSE ALARM RATE OF 0.02 PER SIMULATION AND MIS-DETECTION RATE OF 0.001 PER TARGET

Size	No FPN correction	With FPN correction
1×1	14.85	4.72
1×1.5	11.43	3.48
1×2	9.38	3.10
1.5×1.5	10.49	2.55
2×2	6.35	2.04

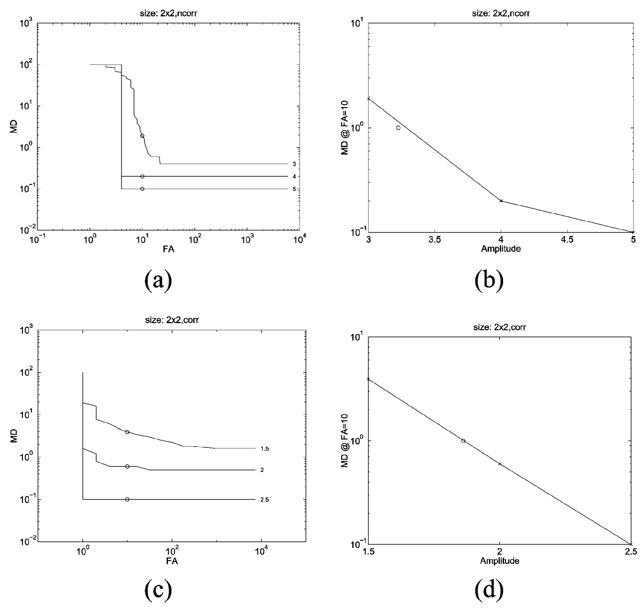


Fig. 8. Results for real noise from camera for 2×2 targets. (a) Plot of MD rate against FA rate (for marked amplitude) for images without FPN correction. The data points are marked as crosses. The MD rate when FA rate is $\mathrm{FA}_T = 10$ per simulation is interpolated, and plotted as circle. (b) Plot of MD rate against amplitude for FA rate of $\mathrm{FA}_T = 10$ per simulation. The data points are marked as crosses. The value of A_T where MD rate is $\mathrm{MD}_T = 0.01$ per target is interpolated and marked as a circle. (c) and (d) Corresponding plots for FPN corrected images.

theoretical analysis suggests that the performance at lower rates of false alarms and misdetections can be extrapolated. Furthermore, due to the normal distribution of noise, even a small increase in the threshold reduces the false alarm and misdetection rates dramatically. Hence, a somewhat higher target amplitude can be expected to reduce these rates to an acceptable level.

In the case of the experiments without FPN correction, the plot of misdetection rate against false alarm rate for different levels of target amplitude is shown in Fig. 8(a). The plot of misdetection rate against SNR for false alarm rate of $\mathrm{FA}_T = 10$ per simulation is shown in Fig. 8(b). The corresponding plots for the experiments with FPN correction are shown in Fig. 8(c) and (d). The target strength required for detection at the specified rates of false alarms and misdetections are marked by circles in Fig. 8(b) and (d). It can be seen that the target strength required when FPN is not corrected $(A_T = 3.22)$ is higher than that required when FPN correction is applied $(A_T = 1.86)$, implying better performance in the latter case.

C. Real Background From an Analog Camera

In this case, a real aerial background, obtained from an analog camera used during a flight test was employed. Targets of size 2×2 pixels were synthetically added for misdetection analysis. In order to suppress the background, low-stop, and morphological preprocessing were separately applied, and the results compared. Since the background was cluttered, a much higher signal was required for satisfactory detection. Even then, the false alarm rate does not reduce sufficiently, thus showing that more postprocessing would be required after applying the algorithm. However, since the number of false alarms (plus true candidates) would be small after this processing, the time complexity of subsequent algorithms would be reduced significantly. The techniques described in [5] can be used to separate the remaining background clutter from the genuine targets. These techniques utilize the difference in the image translation and expansion between an object on a collision course, and the background clutter.

The false alarm threshold was set $\mathrm{FA}_T = 10$ per simulation resulting in a total of ten false alarms for $N_{\mathrm{FA}} = 1$ simulation. The misdetection threshold was $\mathrm{MD}_T = 0.01$ per target, corresponding to ten misdetections for $N_{\mathrm{MD}} = 20$ simulations with $N_{\mathrm{targ}} = 50$ targets. Unfortunately, lower rates for false alarm and misdetection cannot be reliably estimated due to the limited number of background images available.

The results for the morphological filter and the low-stop filter are shown in Fig. 9. It can be seen that the target strength required when the morphological filter $(A_T = 17.8)$ is used is much lower than that required when the low-stop filter $(A_T = 57.8)$ is used. The morphological filter is, thus, better, and the reason for this is that the morphological filter reduces clutter corresponding to large features, whereas the low-stop filter does not do this effectively. However, both result in much poorer performance than that obtained with a digital camera with clear background.

D. Comparison With Other Methods

The performance of the dynamic programming algorithm was compared with other methods such as simple thresholding on a single frame, and temporal averaging on the same number of frames. The comparison was made using FPN correction on images with simulated camera noise. The algorithms used were: dynamic programming algorithm, simple thresholding on a single frame, and temporal averaging on image sequences with moving and stationary targets. Table III shows the comparison for these algorithms using various target sizes. The plots of the threshold amplitudes against target sizes are shown in Fig. 10. Again, smaller threshold amplitudes imply better performance as explained before.

It can be seen that the performance of single frame thresholding, as well as temporal averaging are much poorer than that of the dynamic programming. However, if stationary targets are used instead of moving targets, the performance of temporal averaging is slightly better than that of dynamic programming, showing that temporal averaging is the best choice when the targets are stationary.

VI. Theoretical Performance Characterization

This section theoretically derives the approximate performance of the dynamic programming algorithm, based on the paper by Tonissen and Evans [11]. Temporal averaging and single frame thresholding are taken as special cases of dynamic programming. It should be noted that for theoretical analysis, the recursion step in the dynamic programming algorithm is replaced by

$$F(x,y;u,v;k) = f(x,y;k) + \alpha \max_{(x',y')\in Q} F(x-u-x', y-v-y'; u,v; k-1). \quad (7)$$

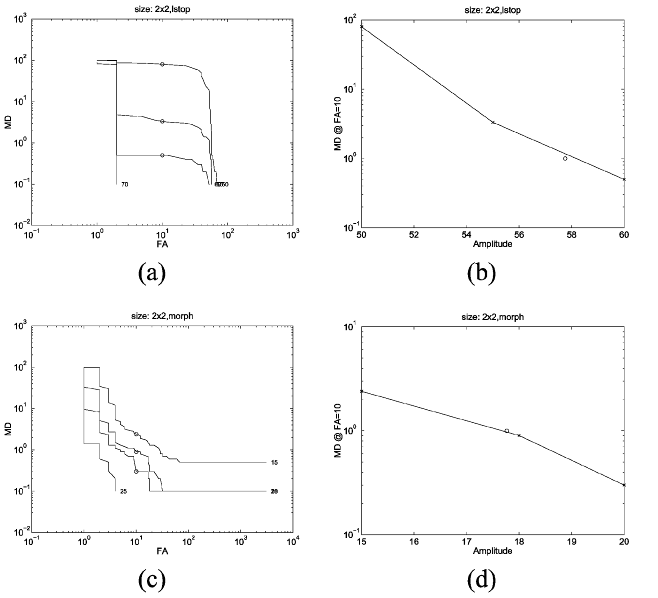


Fig. 9. Results for real cluttered background for 2×2 targets using low stop filter. (a) Plot of MD rate against FA rate (for marked amplitude). The data points are marked as crosses. The MD rate when FA rate is $\mathrm{FA}_T = 10$ per simulation is interpolated, and plotted as circle. (b) Plot of MD rate against amplitude for FA rate of $\mathrm{FA}_T = 10$ per simulation. The data points are marked as crosses. The value of A_T where MD rate is $\mathrm{MD}_T = 0.01$ per target is interpolated and marked as a circle. Corresponding results using morphological preprocessing are shown in (c) and (d).

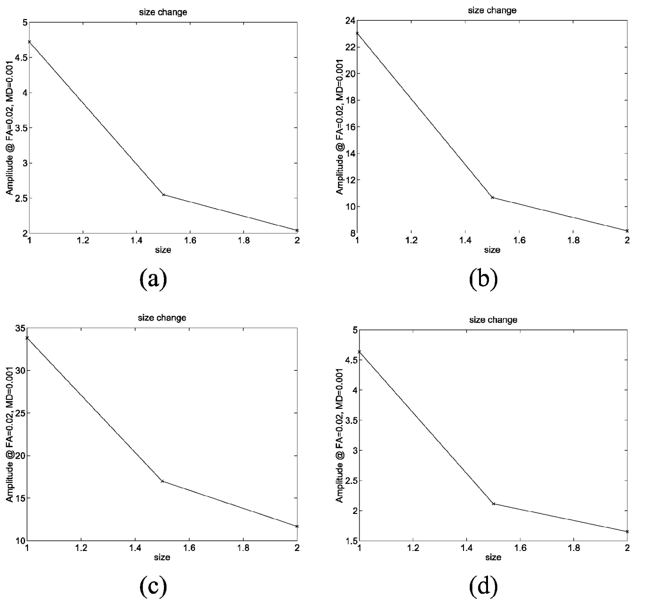


Fig. 10. Performance comparison of several algorithms: Plot of amplitude against the target size $(x \times x)$ for experiments without FPN correction using (a) dynamic programming, (b) thresholding single frame, (c) temporal averaging (moving targets), and (d) temporal averaging (stationary targets).

TABLE III
RESULTS OF TARGET DETECTION ALGORITHMS ON SIMULATED IMAGE SEQUENCES WITH FPN CORRECTION. THRESHOLD AMPLITUDES ARE SHOWN FOR FALSE ALARM RATE OF 0.02 PER SIMULATION AND MISDETECTION RATE OF 0.001 PER TARGET

Size	Dynamic prog.	Single frame thresh.	Temp. Avg. (moving)	Temp. Avg. (stat.)
1×1	4.72	23.03	33.82	4.63
1.5×1.5	2.55	10.68	16.99	2.11
2×2	2.04	8.17	11.67	1.65

However, this only changes F by a scale factor, and since both signal as well as noise would be scaled equally, SNR analysis does not change.

A. False Alarm and Misdetection Probabilities

Probability of false alarms P_{FA} is the probability that there is at least one state exceeding the threshold V_T out of p velocity states at the final output time K, for the pixel where there is no signal in its neighborhood—i.e., hypothesis H_0

$$\begin{aligned} P_{\mathrm{FA}}(x,y) &= Pr\left[\max_{(u,v)\in P} F(x,y;u,v;K) \geq V_T | H_0\right] \\ &= 1 - [P_{0,K}(V_T)]^p \end{aligned} \tag{8}$$

where $P_{0,K}(V_T)$ denotes the probability of F for hypothesis H_0 at time K, being less than or equal to the threshold V_T.

Probability of misdetection P_{MD} is the probability that there is no output with correct velocity (u,v) exceeding the threshold at time K, within a neighborhood R of size $r+1$, where one cell contains signal—i.e., hypothesis H_1—and the other r cells are noise. This allows for some tolerance in the location of target. For example, a 5×5 neighborhood corresponding to $r+1 = 25$ gives a tolerance of ± 2 pixels in the location of the target. On the other hand, a 1×1 neighborhood consisting of only the target position corresponds to $r+1 = 1$ or $r = 0$ giving no tolerance for the target position

$$\begin{aligned} P_{\mathrm{MD}}(x,y;u,v) &= Pr\left[\max_{x',y'\in R} F(x-x',y-y';u,v;K) \right. \\ &\qquad \left. \leq V_T | H_1\right] \\ &= P_{1,K}(V_T)[P_{0,K}(V_T)]^r \end{aligned} \tag{9}$$

where $P_{1,K}(V_T)$ denotes the probability of F for hypothesis H_1 at time K being less than or equal to the threshold V_T.

B. Normal Approximations

For an analytic solution of the performance of the dynamic programming algorithm, the distributions of the intermediate outputs can be approximated using normal approximations. Consider q independent standard normal variables $w_i \sim N(0,1)$. The cumulative distribution function (CDF) of the maximum of these variables is given by

$$P(w) = Pr\left[\max_i w_i \leq w\right] = \prod_i Pr[w_i \leq w] = [\Phi(w)]^q \tag{10}$$

where $\Phi(\cdot)$ is the CDF of a standard normal variable. The probability density function (PDF) is the derivative of the CDF given by

$$p(w) = q[\Phi(w)]^{q-1} G(w) \tag{11}$$

where $G(\cdot)$ is the standard normal PDF.

This distribution of maximum of q standard normal variables can be approximated as a normal distribution $N(\mu_q, \sigma_q^2)$, where μ_q and σ_q^2 denote the mean and the variance of the actual distribution. These are computed using numerical integration, and are tabulated in Table IV for different values of q.

For general normal variables $z_i \sim N(\mu, \sigma^2)$, one can substitute $z_i = \mu + \sigma w_i$, where w_i are standard normal variables. The maximum of z_i is approximately normally distributed with mean and variance given by

$$\begin{aligned} E[\max z_i] &= \mu + \sigma E[\max w_i] = \mu + \sigma\mu_q \\ V[\max z_i] &= \sigma^2 V[\max w_i] = \sigma^2 \sigma_q^2. \end{aligned} \tag{12}$$

Let the input at any time k be normally distributed, in the absence and presence of the target as

$$f(x,y;k|H_0) \sim N(\mu_n, \sigma_n^2), f(x,y;k|H_1) \sim N(\mu_s, \sigma_s^2). \tag{13}$$

Then, the distributions of the output F at time k will be *approximately* normally distributed as

$$\begin{aligned} F(x,y;u,v;k|H_0) &\simeq N(M_{0,k}, S_{0,k}^2) \\ F(x,y;u,v;k|H_1) &\simeq N(M_{1,k}, S_{1,k}^2) \end{aligned} \tag{14}$$

where the M and S parameters are calculated below.

C. False Alarm Analysis

For noise pixels, we have

$$\begin{aligned} F(x,y;u,v;0) &= 0 \\ F(x,y;u,v;k) &= f(x,y;k) + \alpha \max_{(x',y')\in Q} F \\ &\qquad \times (x-u-x', y-v-y'; u,v;k-1) \\ &\simeq N(M_{0,k}, S_{0,k}^2). \end{aligned} \tag{15}$$

Using (12), the mean and variance parameters at time k can be recursively expressed as

$$\begin{aligned} M_{0,0} &= 0, \quad M_{0,k} = \mu_n + \alpha(M_{0,k-1} + \mu_q S_{0,k-1}) \\ S_{0,0}^2 &= 0, \quad S_{0,k}^2 = \sigma_n^2 + \alpha^2 \sigma_q^2 S_{0,k-1}^2. \end{aligned} \tag{16}$$

Solving these recursive equations yields expressions for mean and variance at time K

$$\begin{aligned} S_{0,K}^2 &= \sigma_n^2 \frac{1-\alpha^{2K}\sigma_q^{2K}}{1-\alpha^2\sigma_q^2} \\ M_{0,K} &= \frac{1-\alpha^K}{1-\alpha}\mu_n + \alpha\mu_q \sum_{i=0}^{K-1} (\alpha^i S_{0,K-i-1}). \end{aligned} \tag{17}$$

TABLE IV
VALUES OF μ_q AND σ_q^2 FOR A NUMBER OF VALUES OF q

q	μ_q	σ_q^2
1	0	1
4	1.029	0.491
9	1.485	0.3574
25	1.965	0.2585
49	2.241	0.2168

To get approximate closed-form expressions for $M_{0,K}$, one can write $S_{0,k}$ as

$$S_{0,k} = \sigma_n \sqrt{\frac{1-\alpha^{2K}\sigma_q^{2K}}{1-\alpha^2\sigma_q^2}} \simeq \frac{\sigma_n}{\sqrt{1-\alpha^2\sigma_q^2}}\left(1-\gamma_k\alpha^{2k}\sigma_q^{2k}\right) \tag{18}$$

where γ_k is dependent on k but always lies between 0 and 1. Using $\gamma_k = 1/2$ is equivalent to using the first-order term of binomial expansion, whereas $\gamma_k = 0$ corresponds to assuming that $S_{0,k}$ remains approximately constant with k, which is justifiable, since σ_q^2 is quite small. Accordingly, we have

$$\begin{aligned} M_{0,K} &= \frac{1-\alpha^K}{1-\alpha}\mu_n + \alpha\mu_q \sum_{k=0}^{K-1}(\alpha^k S_{0,K-k-1}) \\ &= \frac{1-\alpha^K}{1-\alpha}\mu_n + \frac{\alpha\mu_q\sigma_n}{\sqrt{1-\alpha^2\sigma_q^2}} \\ &\quad \times \left[\frac{1-\alpha^K}{1-\alpha} - \gamma\alpha^{K-1}\frac{1-(\alpha\sigma_q^2)^K}{1-\alpha\sigma_q^2}\right] \end{aligned} \tag{19}$$

where γ is a function of all γ_k and also lies between 0 and 1. Values of $\gamma = 0$ and $\gamma = 1/2$ can be used as the zero-order and first-order approximations, respectively.

For $K \to \infty$, $\alpha \neq 1$ such that $\alpha^K \ll 1$ (also, $\sigma_q^{2K} \ll 1$), we have

$$\begin{aligned} S_{0,K}^2 &= \frac{\sigma_n^2}{1-\alpha^2\sigma_q^2} \\ M_{0,K} &= \frac{1}{1-\alpha}\left[\mu_n + \frac{\alpha\mu_q\sigma_n}{\sqrt{1-\alpha^2\sigma_q^2}}\right]. \end{aligned} \tag{20}$$

For the case when $\alpha = 1$, the sum $\sum_{i=0}^{K}\alpha^i$ changes from $(1-\alpha^K)/(1-\alpha)$ to K. Hence, the expressions become

$$\begin{aligned} S_{0,K}^2 &= \sigma_n^2\frac{1-\sigma_q^{2K}}{1-\sigma_q^2} \\ M_{0,K} &= K\mu_n + \frac{\mu_q\sigma_n}{\sqrt{1-\sigma_q^2}}\left[K - \gamma\frac{1-(\sigma_q^2)^K}{1-\sigma_q^2}\right]. \end{aligned} \tag{21}$$

Finally, the probability of false alarms is

$$P_{\rm FA} = 1 - [P_{0,K}(V_T)]^p \tag{22}$$

giving

$$P_{0,K}(V_T) = (1-P_{\rm FA})^{1/p} = \Phi\left(\frac{V_T - M_{0,K}}{S_{0,K}}\right) \tag{23}$$

where $\Phi(\cdot)$ denotes the CDF of a standard normal variable. Hence, the threshold V_T can be expressed in terms of the mean $M_{0,K}$, variance $S_{0,K}$, and the false alarm probability $P_{\rm FA}$ as

$$V_T = M_{0,K} + S_{0,K}\Phi^{-1}[(1-P_{\rm FA})^{1/p}] = M_{0,K} + S_{0,K}\phi_{0,p} \tag{24}$$

where

$$\phi_{0,p} = \Phi^{-1}[(1-P_{\rm FA})^{1/p}] \simeq \Phi^{-1}\left[1 - \frac{P_{\rm FA}}{p}\right]. \tag{25}$$

D. Misdetection Analysis

The probability of misdetection is given by

$$P_{\rm MD} = P_{1,K}(V_T)[P_{0,K}(V_T)]^r \le P_{1,K}(V_T). \tag{26}$$

Substituting the expression of V_T in terms of false alarm rate, we have

$$P_{\rm MD} = (1-P_{\rm FA})^{r/p}P_{1,K}(V_T) \tag{27}$$

giving

$$P_{1,K}(V_T) = \frac{P_{\rm MD}}{(1-P_{\rm FA})^{r/p}} = \Phi\left(\frac{V_T - M_{1,K}}{S_{1,K}}\right). \tag{28}$$

Hence

$$\begin{aligned} V_T &= M_{1,K} + S_{1,K}\Phi^{-1}\left[\frac{P_{\rm MD}}{(1-P_{\rm FA})^{r/p}}\right] \\ &= M_{1,K} - S_{1,K}\phi_{1,p} \end{aligned} \tag{29}$$

where

$$\begin{aligned} \phi_{1,p} &= -\Phi^{-1}\left[\frac{P_{\rm MD}}{(1-P_{\rm FA})^{r/p}}\right] \\ &= \Phi^{-1}\left[1 - \frac{P_{\rm MD}}{(1-P_{\rm FA})^{r/p}}\right] \simeq \Phi^{-1}[1-P_{\rm MD}] \end{aligned} \tag{30}$$

since, usually, $P_{\rm FA} \ll 1$.

Approximations of $M_{1,K}$ and $S_{1,K}^2$ are obtained considering the exceeding of threshold only due to the signal part and not due to the noise part. Also, it is assumed that the target occupies a single pixel. In such a case, we have

$$\begin{aligned} F(x,y;u,v;k) &\simeq f(x,y;k) + \alpha F(x,y;u,v;k-1) \\ &\simeq N(M_{1,k}, S_{1,k}^2). \end{aligned} \tag{31}$$

It can be easily shown that

$$M_{1,K} \simeq \frac{1-\alpha^K}{1-\alpha}\mu_s, \quad S_{1,K}^2 \simeq \frac{1-\alpha^{2K}}{1-\alpha^2}\sigma_s^2. \tag{32}$$

E. Calculation of Required SNR

To calculate the SNR required for detection at particular rates of false alarms and misdetections, (24) and (29) are combined to give

$$M_{1,K} - M_{0,K} = S_{0,K}\phi_{0,p} + S_{1,K}\phi_{1,p}. \tag{33}$$

Using expressions for $S_{0,K}$, $M_{0,K}$, $S_{1,K}$, and $M_{1,K}$, and assuming $\mu_n = 0$, $\mu_s = \mu$, and $\sigma_n = \sigma_s = \sigma$, the SNR required for detection is obtained as

$$\begin{aligned}\mathrm{SNR}_T &= \frac{\mu}{\sigma}\\ &\simeq \frac{\alpha\mu_q}{\sqrt{1-\alpha^2\sigma_q^2}}\left[1-\gamma\alpha^{K-1}\frac{1-\alpha}{1-\alpha^K}\cdot\frac{1-\alpha^K\sigma_q^{2K}}{1-\alpha\sigma_q^2}\right]\\ &\quad+\frac{1-\alpha}{1-\alpha^K}\sqrt{\frac{1-\alpha^{2K}\sigma_q^{2K}}{1-\alpha^2\sigma_q^2}}\phi_{0,p}\\ &\quad+\sqrt{\frac{1-\alpha}{1+\alpha}\cdot\frac{1+\alpha^K}{1-\alpha^K}}\phi_{1,p}.\end{aligned} \tag{34}$$

For $\alpha = 1$, replacing $(1-\alpha^K)/(1-\alpha)$ by K, we get

$$\begin{aligned}\mathrm{SNR_T} &= \frac{\mu}{\sigma}\\ &\simeq \frac{\mu_q}{\sqrt{1-\sigma_q^2}}\left[1-\frac{\gamma}{K}\cdot\frac{1-\sigma_q^{2K}}{1-\sigma_q^2}\right]\\ &\quad+\frac{1}{K}\sqrt{\frac{1-\sigma_q^{2K}}{1-\sigma_q^2}}\phi_{0,p}+\frac{1}{\sqrt{K}}\phi_{1,p}.\end{aligned} \tag{35}$$

For $K \to \infty$, $\alpha \neq 1$ such that $\alpha^K \ll 1$

$$\mathrm{SNR}_T = \frac{\mu}{\sigma} \simeq \frac{\alpha\mu_q}{1-\alpha^2\sigma_q^2} + \frac{1-\alpha}{\sqrt{1-\alpha^2\sigma_q^2}}\phi_{0,p} + \sqrt{\frac{1-\alpha}{1+\alpha}}\phi_{1,p}. \tag{36}$$

The above expressions of SNR_T can be written in the form

$$\mathrm{SNR}_T = A + B\phi_{0,p} + C\phi 1,p \tag{37}$$

where A, B, and C depend on K, q, and α. The terms B and C decrease with K, improving the algorithm performance as K increases. However, the term A *increases* with K, putting a lower bound on the *required* SNR, thus limiting the performance. It can be shown that this bound increases with q, and, hence, a lowest possible value of q should be used. This is intuitively explained, since a maximum is taken over q noise pixels and it is more likely to be a false alarm when q is large.

F. Temporal Averaging and Single Frame Thresholding as Special Cases

For detection of stationary targets, the optimal detector is given by pixelwise temporal averaging. However, direct use of temporal averaging results in infinite memory. To give a higher weight to more recent observations, the following recursive filter can be used:

$$F(x,y;0) = 0, F(x,y;k) = f(x,y;k) + \alpha F(x,y;k-1). \tag{38}$$

This recursive temporal averaging can be considered special case of dynamic programming with $p = q = 1$, for which $\mu_q = 0$ and $\sigma_q^2 = 1$. Hence, the threshold SNR for recursive temporal averaging becomes

$$\mathrm{SNR}_T = \frac{\mu}{\sigma} \simeq \sqrt{\frac{1-\alpha}{1+\alpha}\cdot\frac{1+\alpha^K}{1-\alpha^K}}\left[\phi_{0,1}+\phi_{1,1}\right]. \tag{39}$$

Also, for $\alpha = 1$, this expression takes the limit

$$\mathrm{SNR}_T = \frac{1}{\sqrt{K}}\left[\phi_{0,1}+\phi_{1,1}\right]. \tag{40}$$

The same result would be obtained by using $\alpha = 1$ in original equations. For $K \to \infty$, $\alpha \neq 1$ such that $\alpha^K \ll 1$

$$\mathrm{SNR}_T = \sqrt{\frac{1-\alpha}{1+\alpha}}\left[\phi_{0,1}+\phi_{1,1}\right]. \tag{41}$$

For single-frame thresholding ($K = 1$ or $\alpha = 0$), the threshold SNR reduces to $\phi_{0,1} + \phi_{1,1}$.

Note that the first term from the dynamic programming algorithm disappears in these expressions, and there is no lower limit to the performance if $\alpha = 1$.

G. Analysis

Fig. 11(a) shows plots of SNR_T against K for the dynamic programming algorithm with $p = q = 4$ and a number of values of α. The false alarm rate is 2×10^{-8} (0.02 per simulation for a $1K \times 1K$ image), and the misdetection rate is 0.001. It can be seen that SNR_T decreases with increase in K, but saturates at a certain point depending on α. It should be noted that lower *required* SNR means *better* performance. Fig. 11(b) shows the corresponding plot for $p = q = 1$—i.e., recursive temporal averaging. Fig. 11(c) and (d) shows the plots of SNR_T against K with $\alpha = 1$ and $\alpha = 15/16$, respectively, for a number of values of p and q. It is observed that SNR_T increases with q as expected. The SNR_T also increases slightly with p, but the plots cannot show the change. Except in the case of $\alpha = 1$ and $p = q = 1$—i.e., temporal averaging—the SNR_T saturates at some minimum value as $K \to \infty$.

H. Comparison Between Theoretical and Observed Performance

The parameters used in the calculation of theoretical performance of the algorithms for 2×2 targets are shown in Table V. The calculated and the observed SNR threshold for these parameters for various algorithms are shown in Table VI.

One can observe that the actual performance of the algorithm for 2×2 targets is slightly better than the theoretical performance for most of the algorithms. The reason for this is, that a 2×2 target occupies at least one pixel completely, and a few other pixels partially. Hence, its performance should be slightly greater than the calculated performance in which one assumes that the target occupies exactly one pixel.

To correct this problem, point targets were used in place of 2×2 targets. The experiments described in previous sections were repeated using point targets. The comparison between the calculated and observed SNR for a number of false alarm and misdetection rates are shown in Table VII. It can be seen that

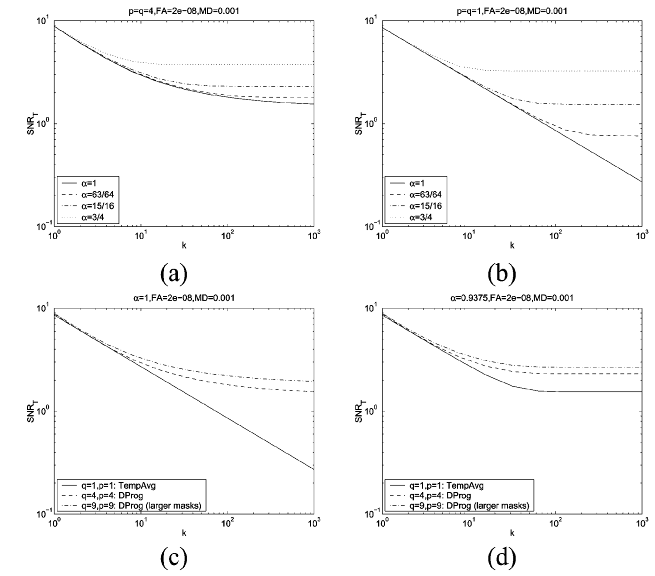


Fig. 11. Plots of SNR_T against K for (a) $p = q = 4$ (dynamic programming) and number of α values; (b) $p = q = 1$ (temporal averaging) and number of α values; (c) $\alpha = 1$ and number of p and q values; and (d) $\alpha = 15/16$ and number of p and q values. The parameters used are $\mathrm{FA} = 2 \times 10^{-8}$, $\mathrm{MD} = 0.001$.

TABLE V
PARAMETERS USED FOR CALCULATING THE THEORETICAL PERFORMANCE OF ALGORITHMS

Parameter	Dynamic prog	Single frame	Temp. Avg. (stat)
FA	$2 \times 10^{-8}/pixel = 0.02/image$		
MD	$0.001/pixel$		
α	$15/16$		
K	32	1	32
q	4	—	1

TABLE VI
COMPARISON OF THEORETICAL PERFORMANCE OF THE ALGORITHMS WITH OBSERVED PERFORMANCE ON 2 × 2 TARGETS

Algorithm	Theoretical SNR	Observed SNR = A_T/σ_{noise}
Dyn. Prog	2.4540	2.07
Single frame	8.5811	8.29
Temp. Avg. (stat.)	1.7507	1.67

the calculated and observed SNR rates agree very well in most cases. However, in the case of extremely low false alarm and misdetection rates, the observed SNR is greater than the calculated SNR for the dynamic programming algorithm. The reason for this is the normal approximation used for the distribution of the resulting output.

VII. CONCLUSION

This paper described a protocol and a set of experiments to characterize and compare the performance of target detection algorithms. The experimental protocol described how to generate targets, model noise and obtain the ROC—i.e., plots of missed detection rates against the false alarm rates. The methodology presented in [8] was used to combine multiple ROC into a few performance curves, showing the variation of minimum target amplitude required against a target parameter (such as the size of object), for fixed false alarm and missed detection rates. It was observed that smaller target amplitude is required for larger target size, as expected. Other parameters such as, whether the FPN is corrected or not, for simulated as well as real digital

TABLE VII
COMPARISON OF THEORETICAL PERFORMANCE OF THE ALGORITHMS WITH OBSERVED PERFORMANCE ON POINT TARGETS FOR A NUMBER OF DIFFERENT VALUES OF FALSE ALARM (FA) AND MISDETECTION (MD) RATES

Algorithm	FA rate	MD rate	Theo. SNR	Obs. SNR
Dyn. Prog.	2×10^{-8}=0.02/sim	0.001	2.4540	2.76
Dyn. Prog.	10^{-6}=1/sim	0.01	2.2313	2.33
Dyn. Prog.	10^{-6}=1/sim	0.1	2.0181	2.02
Dyn. Prog.	10^{-4}=100/sim	0.1	1.9259	1.87
Temp. Avg.	2×10^{-8}=0.02/sim	0.001	1.7507	1.76
Temp. Avg.	10^{-6}=1/sim	0.01	1.4444	1.45
Temp. Avg.	10^{-6}=1/sim	0.1	1.2313	1.25

camera images without clutter, were also varied. It is concluded that the fixed pattern noise correction improves the performance of the algorithm as expected, in both simulated and real camera. This shows that the FPN correction should be performed to improve the target detection performance. For real images with clutter, the use of morphological and low-stop filtering for the preprocessing step were compared. It was observed that the morphological filter is more effective than the low-stop filter for removal of large clutter.

A comparison between methods show that dynamic programming algorithm performs much better than thresholding a single frame to detect targets, while in the case of stationary targets, temporal averaging of frames performs slightly better than dynamic programming, although temporal averaging is significantly faster than the dynamic programming algorithm. However, the performance of temporal averaging severely degrades for moving targets.

The theoretical expressions for the SNR required for detection were derived using statistical methods. It was observed that the experimental results agreed well with the theoretical values.

REFERENCES

[1] J. Arnold, S. Shaw, and H. Pasternack, "Efficient target tracking using dynamic programming," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 29, no. 1, pp. 44–56, Jan. 1993.

[2] Y. Barniv, "Dynamic programming solution for detecting dim moving targets," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 21, no. 1, pp. 144–156, Jan. 1985.

[3] D. Casasent and A. Ye, "Detection filters and algorithm fusion for ATR," *IEEE Trans. Image Process.*, vol. 6, no. 1, pp. 114–125, Jan. 1997.

[4] T. Gandhi, "Image Sequence Analysis for Object Detection and Segmentation," Ph.D. dissertation, Dept. Comput. Sci. Eng., Pennsylvania State Univ., University Park, 2000.

[5] T. Gandhi, M.-T. Yang, R. Kasturi, O. Camps, L. Coraor, and J. McCandless, "Detection of obstacles in the flight path of an aircraft," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, vol. 2, Mar. 2000, pp. 304–311.

[6] G. E. Healey and R. Kondepudy, "Radiometric CCD camera calibration and noise estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 3, pp. 267–276, Mar. 1994.

[7] G. C. Holst, *CCD Arrays, Cameras and Displays*. Winter Park, FL: JCD, 1996.

[8] T. Kanungo, M. Y. Jaisimha, J. Palmer, and R. M. Haralick, "A methodology for quantitative performance evaluation of detection algorithms," *IEEE Trans. Image Process.*, vol. 4, no. 12, pp. 1667–1674, Dec. 1995.

[9] R. Kasturi, O. Camps, L. Coraor, T. Gandhi, M.-T. Yang, and R. Kasturi, "Obstacle Detection Algorithms for Aircraft Navigation," Dept. Comput. Sci. Eng., Pennsylvania State Univ., University Park, Tech. Rep. CSE-00-002, 2000.

[10] K. Nishiguchi, M. Kobayashi, and A. Ichikawa, "Small target detection from image sequences using recursive max filter," in *Proc. SPIE*, vol. 2561, Jul. 1995, pp. 153–166.

[11] S. M. Tonissen and R. J. Evans, "Performance of dynamic programming techniques for track-before-detect," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 32, no. 4, pp. 1440–1451, Oct. 1996.

Tarak Gandhi (S'93–M'00) received the B.Tech. degree in computer science and engineering from the Indian Institute of Technology, Bombay, and the M.S. and Ph.D. degrees in computer science and engineering (specializing in computer vision, which he applied to detecting objects from an aircraft to prevent accidents) from the Pennsylvania State University, University Park.

He was with Adept Technology, Inc, where he worked on designing algorithms for robotic systems. Currently, he is Postdoctoral Scholar at the Computer Vision and Robotics Research Laboratory, University of California at San Diego, La Jolla. His interests include computer vision, motion analysis, image processing, robotics, target detection, and pattern recognition. He is working on projects involving intelligent driver assistance, motion-based event detection, traffic flow analysis, and structural health monitoring of bridges.

Mau-Tsuen Yang received the B.S. degree in applied mathematics from the National Tsing-Hua University, Taiwan, R.O.C., in 1992, and the Ph.D. degree in computer science and engineering from the the Pennsylvania State University, University Park, in 2000.

He is currently an Assistant Professor with the Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien, Taiwan. His research interests include computer vision, virtual reality, and image-based rendering.

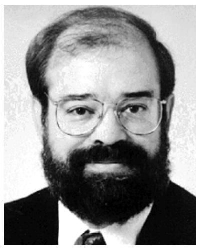
Rangachar Kasturi (S'82–M'82–SM'88–F'96) received the Ph.D. degree from Texas Tech University, Lubbock, in 1982.

He is the Douglas W. Hood Professor and Chair of the Computer Science and Engineering Department, University of South Florida, Tampa. He was a Professor of Computer Science and Engineering and Electrical Engineering at the Pennsylvania State University, University Park, from 1982 to 2003. He is an author of the textbook *Machine Vision* (McGraw-Hill, 1995) and has published numerous papers and research reference books. His research interests are in computer vision and pattern recognition.

Dr. Kasturi is a Fellow of IAPR. He served as the President of the International Association for Pattern Recognition (IAPR) from 2002 to 2004. He was a Fulbright scholar in 1999. He served as Editor-in-Chief of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE and the *Machine Vision and Applications* journals.

Octavia I. Camps (S'89–M'91) received the B.S. degree in computer science and the B.S. degree in electrical engineering from the Universidad de la Republica, Montevideo, Uruguay, in 1981 and 1984, respectively, and the M.S. and Ph.D. degrees in electrical engineering from the University of Washington, Seattle, in 1987 and 1992, respectively.

In 1991, she joined the faculty of the Pennsylvania State University, University Park, where she currently is an Associate Professor with the Departments of Electrical Engineering and Computer Science and Engineering. In 2000, she was a Visiting Faculty at the California Institute of Technology, Pasadena, and at the University of Southern California. Her current research interests includerobust computer vision, image processing, and pattern recognition.



Lee D. Coraor (S'70–M'78) received the B.S. degree in electrical engineering from the Pennsylvania State University (Penn State), University Park, in 1974, and the Ph.D. in electrical engineering from the University of Iowa, Ames, in 1978.

He was a member of the faculty at The Southern Illinois University, Carbondale, from August 1978 to August 1980. He then joined the Department of Electrical Engineering, Penn State, where he is currently an Associate Professor in the Department of Computer Science and Engineering. His current research interests include reconfigurable computing applications, intelligent memory designs, computer architecture, and digital systems. His recent projects have included the use of reconfigurable FPGAs for implementing event-driven simulation; implementation of special purpose hardware/software on aircraft for real time collision avoidance; the design and implementation of a dual computer control system for a micro-gravity continuous flow electrophoresis system; and the development of the SmartDIMM Platform for use as a reconfigurable System-On-Chip prototype.

Jeffrey McCandless received the M.S. degree in mechanical engineering and the Ph.D. degree in vision science from the University of California, Berkeley.

Since 1996, he has been with the NASA Ames Research Center, Moffett Field, CA. For the first part of his career at the Ames Research Center, he developed image-processing algorithms for the High Speed Research Program. He also investigated the perceptual effects of time-delay in head-mounted displays. More recently, he helped design updated displays for the cockpit of the Space Shuttle. He is an instrument-rated private pilot.